1. Lottery Schedule
   (a) What is the main problem with Lottery Scheduler? Explain why it is a problem
   The main problem is it is not possible to guarantee anything in lottery scheduler, it's all probability and expectations. The issue is over any fixed time window, there will be some variance in both CPU time and response time due to the probabilistic nature of the mechanism, and the absolute value of this deviation grows over time. This lead to Stride Scheduling (follow up paper by same authors) which is a deterministic lottery scheduler.

   (b) At a high level, as a mechanism how does lottery scheduling provide modular resource management (allow different modules to control their own internal scheduling)? Why can't the Unix scheduler provide a similar kind of management?
   Lottery Scheduler represents resource rights as lottery tickets which provide a building block for modular resource management. Since each ticket gives a probabilistically guaranteed right to its owner to a worst-case resource consumption rate, each ticket can be used to insulate the resource management policies of independent modules. The policies include ticket transfers, ticket inflation, ticket currencies and compensation tickets. Unix uses a priority-based scheduler which is poorly understood because assigning priorities and dynamically adjusting them is very hard to get right.

   (c) We have two threads A and B that have the same amount of base tickets say 400 each. The time quantum each thread gets when running is 100 ms and both threads need to run for 120 ms. Is there any way for thread A to increase its chance of finishing before thread B? (Hint: Think about compensation tickets)
   There is a way if thread B always consumes all of the 100 ms time quantum while thread A uses 20 ms or 1/5 of the time quantum and then the full 100 ms time quantum. This leads to thread A having 5 times more chance of running again after its first 20 ms run and since it will use the whole time quantum on its second run it will finish faster even though A and B are supposed to have an equal 1 to 1 ratio since they have the same amount of tickets.

2. Mach and L3/L4
   (a) An unusual feature of Mach is the blending of memory and interprocess-communication features, Mach allows each to be used in the implementation of the other. Provide at least two advantages that Mach gains from the blend of memory and IPC.
   Memory objects are represented by ports and IPC are messages sent to this port to request operations on the object. Since IPC are used memory objects may reside on remote systems and be accessed transparently.
   Mach uses virtual-memory remapping to transfer the contents of large messages. Basically Mach modifies the receiving task's address space to include a copy of the message contents. This has improved performance over UNIX message passing.

   (b) The L3 kernel only implements 3 functions for address space: mapping, flushing and granting. Mapping and flushing are required to implement memory managers and pages on top of the u-kernel but is granting also required for a memory manager to work? Explain
   No it is not required for a memory manager to work and it is used only in special situations to avoid double bookkeeping and address-space overflow. It is basically used for convenience and improving performance.
(c) Imagine we have a HTTP web server running in user space. For Mach and L3, consider the path a network packet containing an HTTP request takes as it travels from the network interface card to a Web server process running at user level. Assume that the web server process is separated from the Linux/BSD operating system running in user space. Describe in high level language for example, for Linux: Linux kernel -> [interrupt] -> web server process

Assuming L3 is running Linux on it which L4 does do.
MACH: The incoming packet steps are Mach kernel -> [IPC] -> BSD -> [IPC] -> Mach kernel -> web server process.

3. Nooks
(a) Give one example of a type of bug that Nooks does not deal with. Why does it not deal with it?
Nooks cannot prevent extensions from deliberately executing privileged instructions that corrupt system state. This is because Nooks runs extensions in kernel mode for backward compatibility.

(b) Louis Reasoner is working on his computer that runs Nooks. He decides to print a classified document from his computer to the printer in the room next door. Louis finds it suspicious that it takes unusually longer for him to receive a completed print job notification for document size. Concerned about someone stealing pages of his document he runs to the printer and finds that all his pages are there. Is Louis safe from anyone stealing a page of his document? Assume that someone was hovering over the printer while it was printing and their only means of stealing the document is actually taking a printed page.
The printer driver may have crashed half way through printing the document, causing Nooks to restart the driver and reprinting the whole document because it lost the state that it was in. Nooks tries to do this transparently without the application finding out. Therefore someone hovering over the printer could of stolen the printed pages before the crash without Louis knowing about it.

4. Exokernel
(a) The Exokernel tries to separate protection from management, meaning they protect resources but delegate management to applications. Give an example of a resource and how the exokernel applies this principle to the resource. (hint: provide a resource and state how the application manages it and how the exokernel protects it)
Each application manages its own disk-block cache but the Exokernel allows cached pages to be shared securely across all applications. Thus the exokernel protects pages and disk blocks, but applications manage them.

(b) List two problems that will dissuade Microsoft from creating an exokernel operating system for wide spread adoption. Explain (hint: think about 3rd party companies developing for an exokernel system)
Portability - Applications are created for a specific resource.
Applications are more difficult to write - application now have to deal with managing the resources on the system. Every application has to have knowledge of the resource it runs on. Libraries sort of help with this problem.
Protection - can't trust that all applications are bug free.